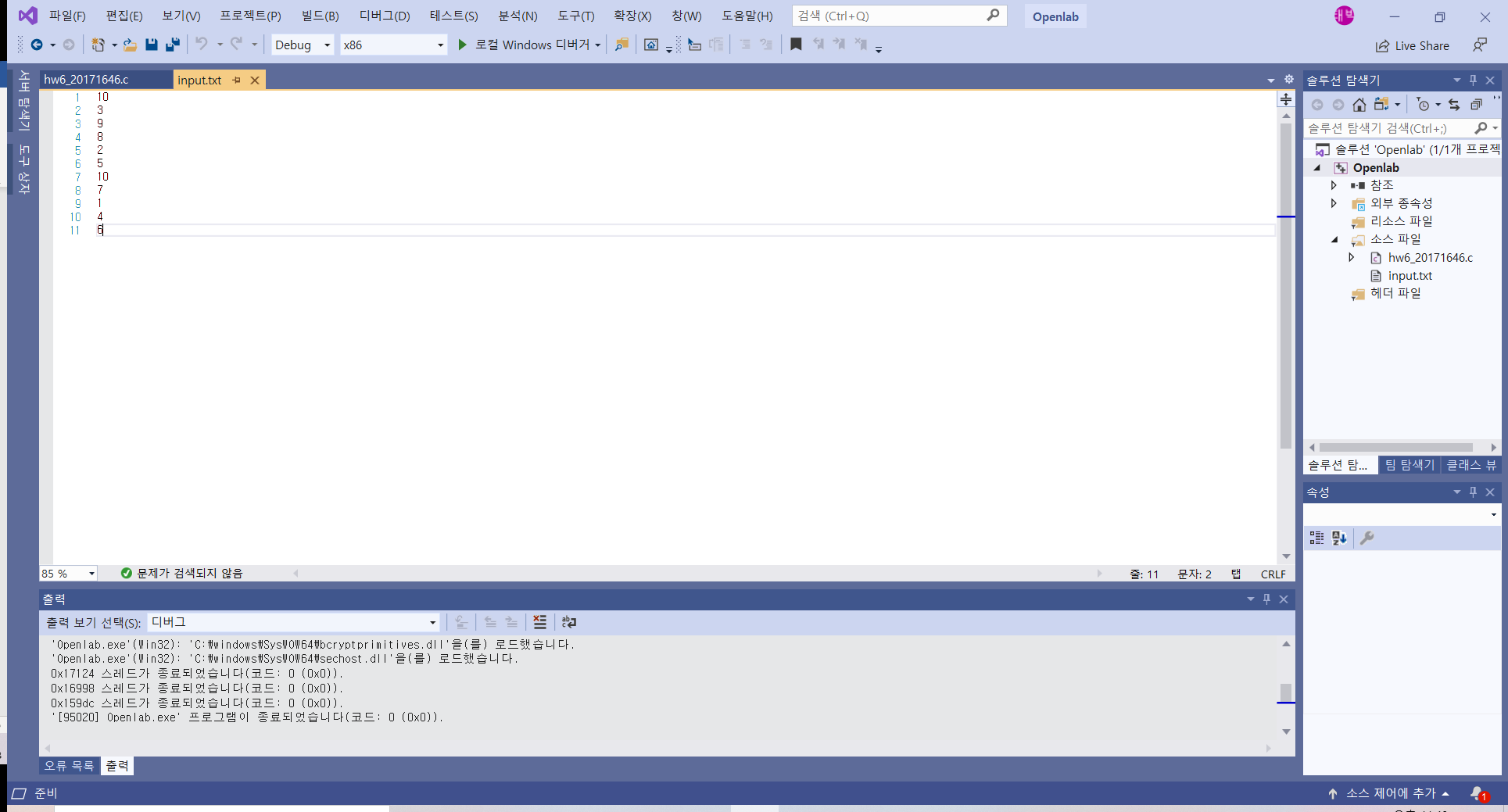
자료구조 오픈랩6

컴퓨터공학과 20171646 박태윤

**- 코드 실행 결과**



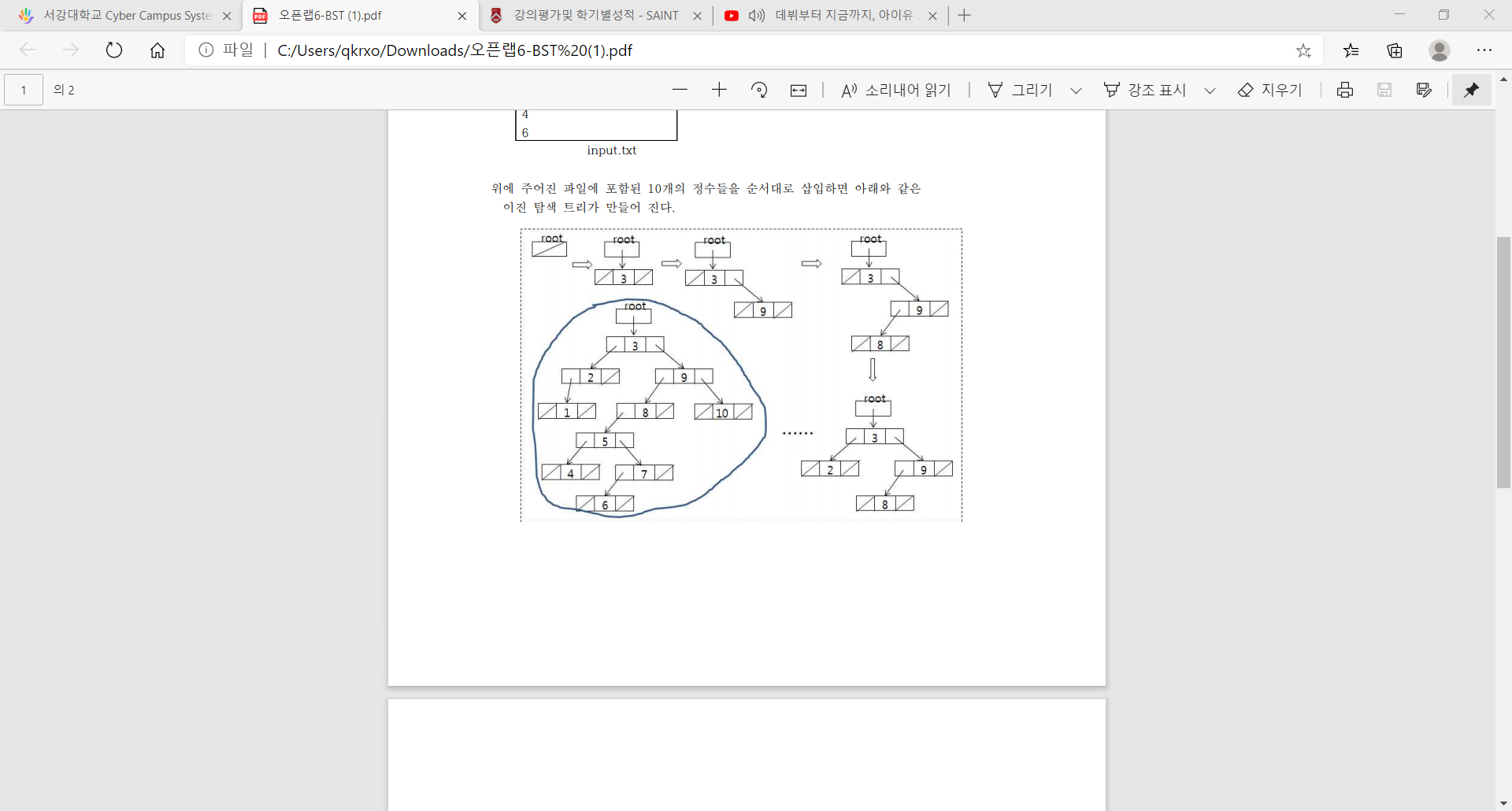
위와 같이 input.txt를 입력으로 받아서 Binary Search Tree를 생성하고 이를 inorder, preorder로 탐색한 결과를 출력해주는 hw6\_20171646.c코드를 실행하였다.

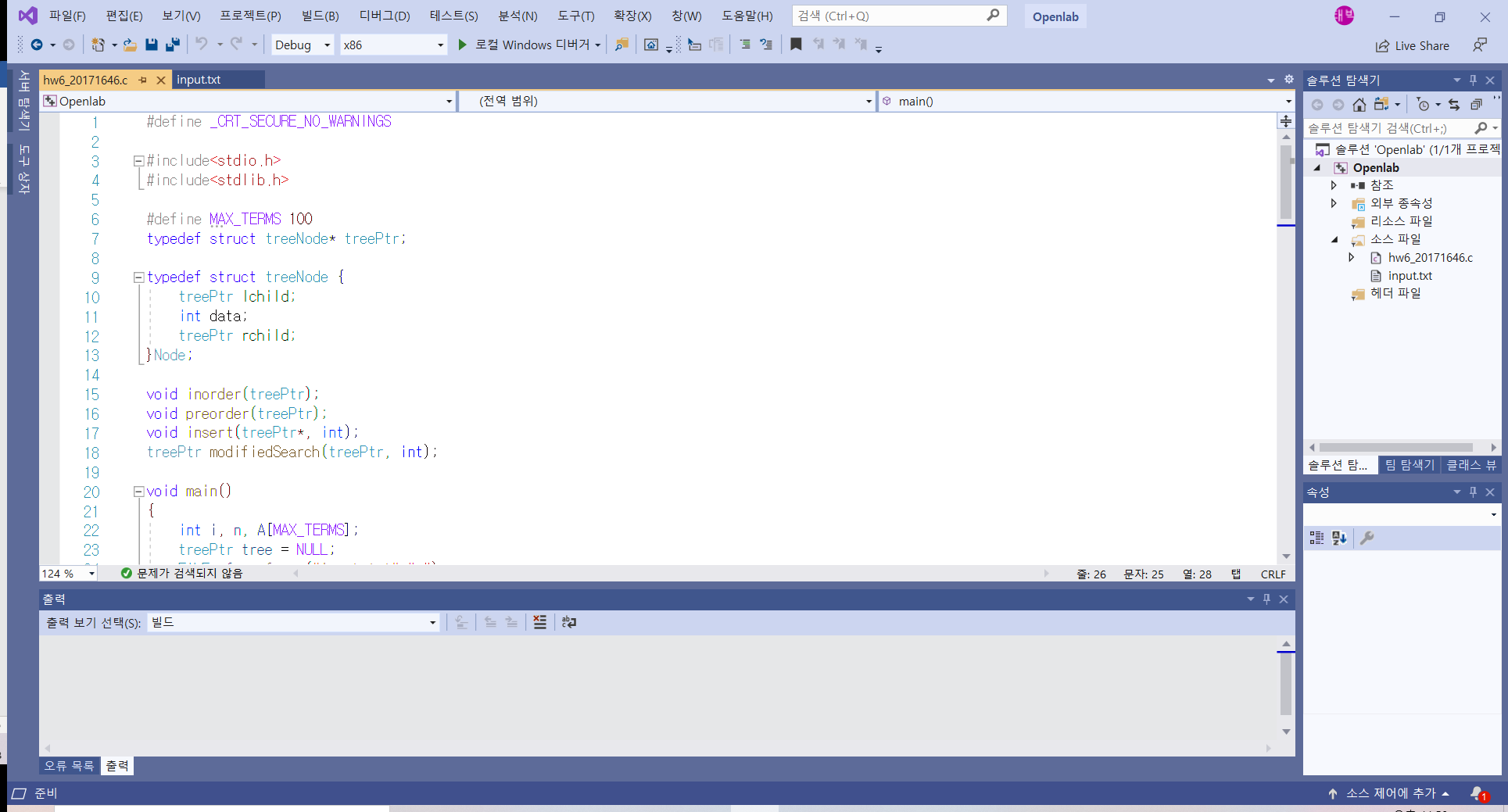
텍스트이(가) 표시된 사진

자동 생성된 설명

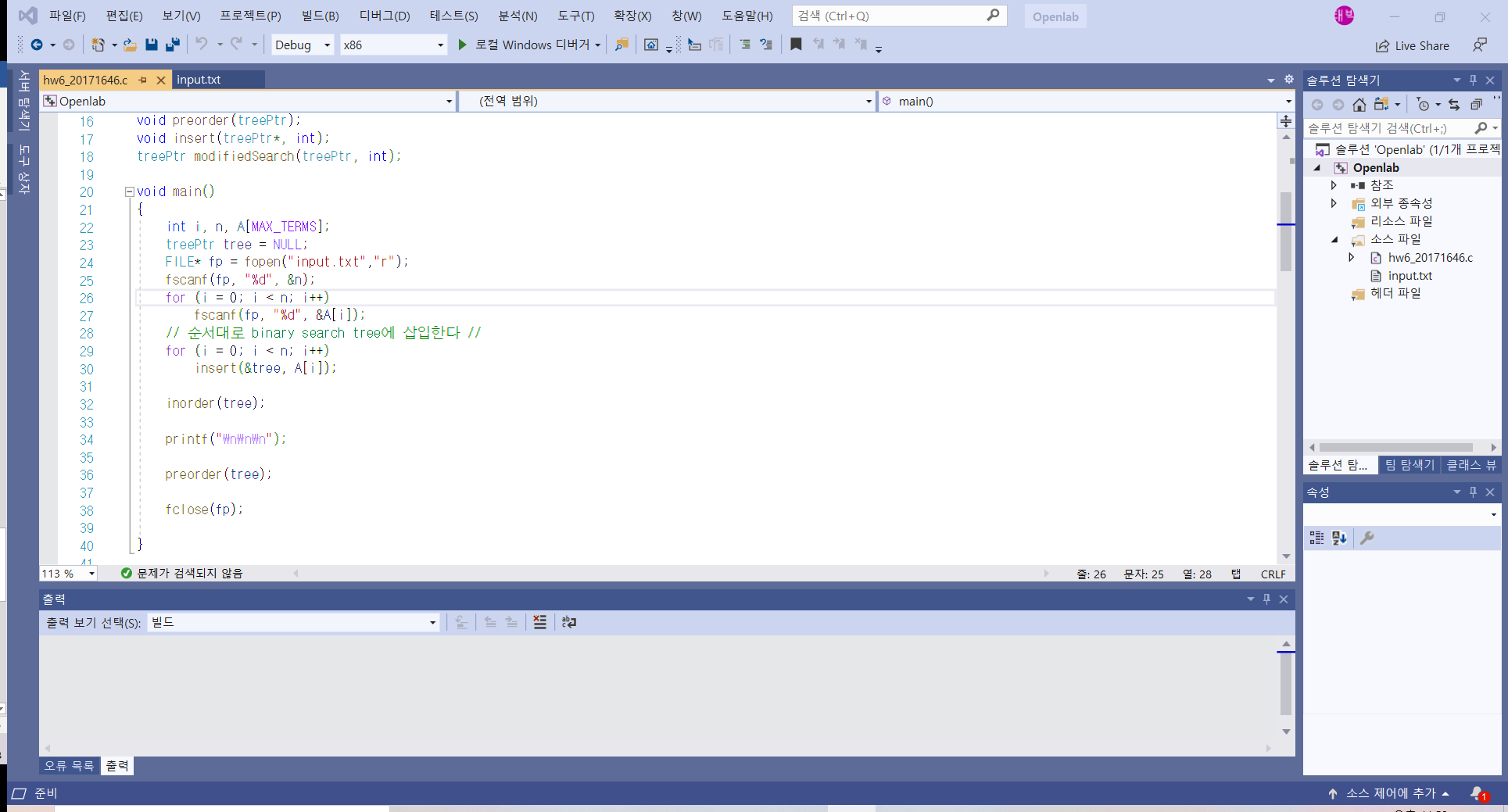
**- 코드 & 알고리즘 설명**

입력 파일에서 첫 줄에 데이터의 개수가 주어진다. 이후 데이터들이 주어지는데, 첫 데이터부터 시작하여(root) 데이터 간 크기를 비교해가며 트리를 만들어준다. 부모 노드의 왼쪽 자식 노드는 부모 노드보다 무조건 작은 데이터여야 하며 오른쪽 자식 노드는 무조건 값이 큰 데이터여야 한다. 위의 input.txt를 사용하여 BST를 만들면 3(root)부터 시작하여 트리를 구성한다. 최종적으로 다음 그림과 같이 트리를 구성할 수 있다.

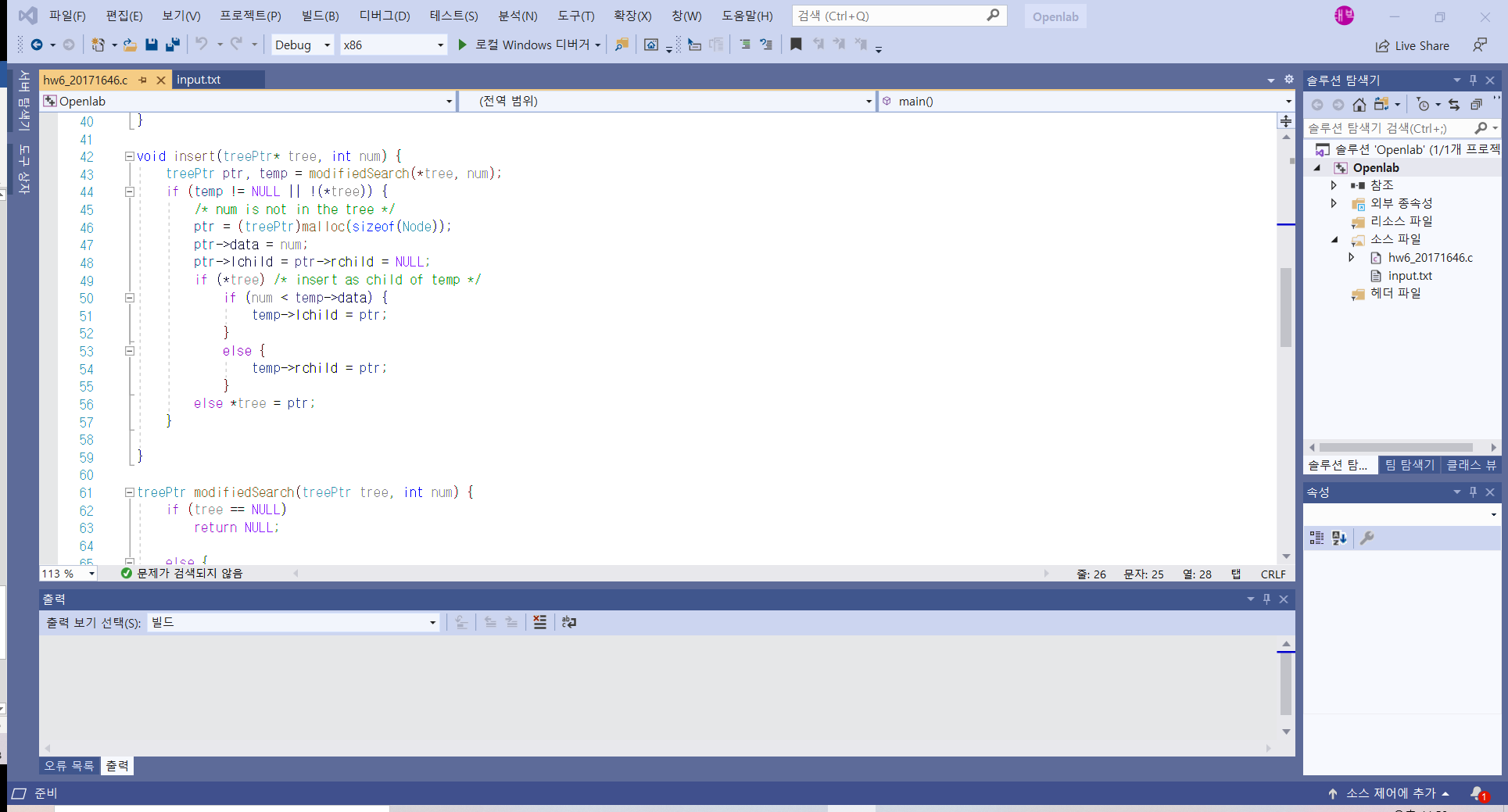




tree의 노드로 링크 lchild와 rchild를 가지며 int형 data를 가지는 Node를 선언하였다. 4개의 함수를 추가적으로 선언하였는데, inorder는 트리를 inorder로 탐색한 결과를 출력해주는 함수이다. preorder는 트리를 preorder로 탐색한 결과를 출력해주는 함수이며 insert는 트리를 만들어주는 함수이고 마지막으로 modifiedSearch는 트리를 만들어줄 때 특정 노드를 트리에 삽입할 수 있는지, 삽입할 수 있다면 삽입해야 하는 위치의 부모 노드를 알려주는 함수이다.



위와 같이 메인 함수를 선언하였다. 처음에 treePtr tree를 선언하여 NULL로 초기화 했으며 “input.txt”파일을 불러왔다. 텍스트 파일 첫 줄에 쓰여진 값을 n에 저장하였는데 n은 노드의 총 개수를 의미한다. 이후 순서대로 fscanf하여 int형 배열 A에 모두 저장하였다. 이후 노드의 총 개수 n만큼 insert함수를 호출하여 트리를 구성해주었으며 inorder, preorder함수를 호출하여 각 탐색의 결과를 출력하였다.

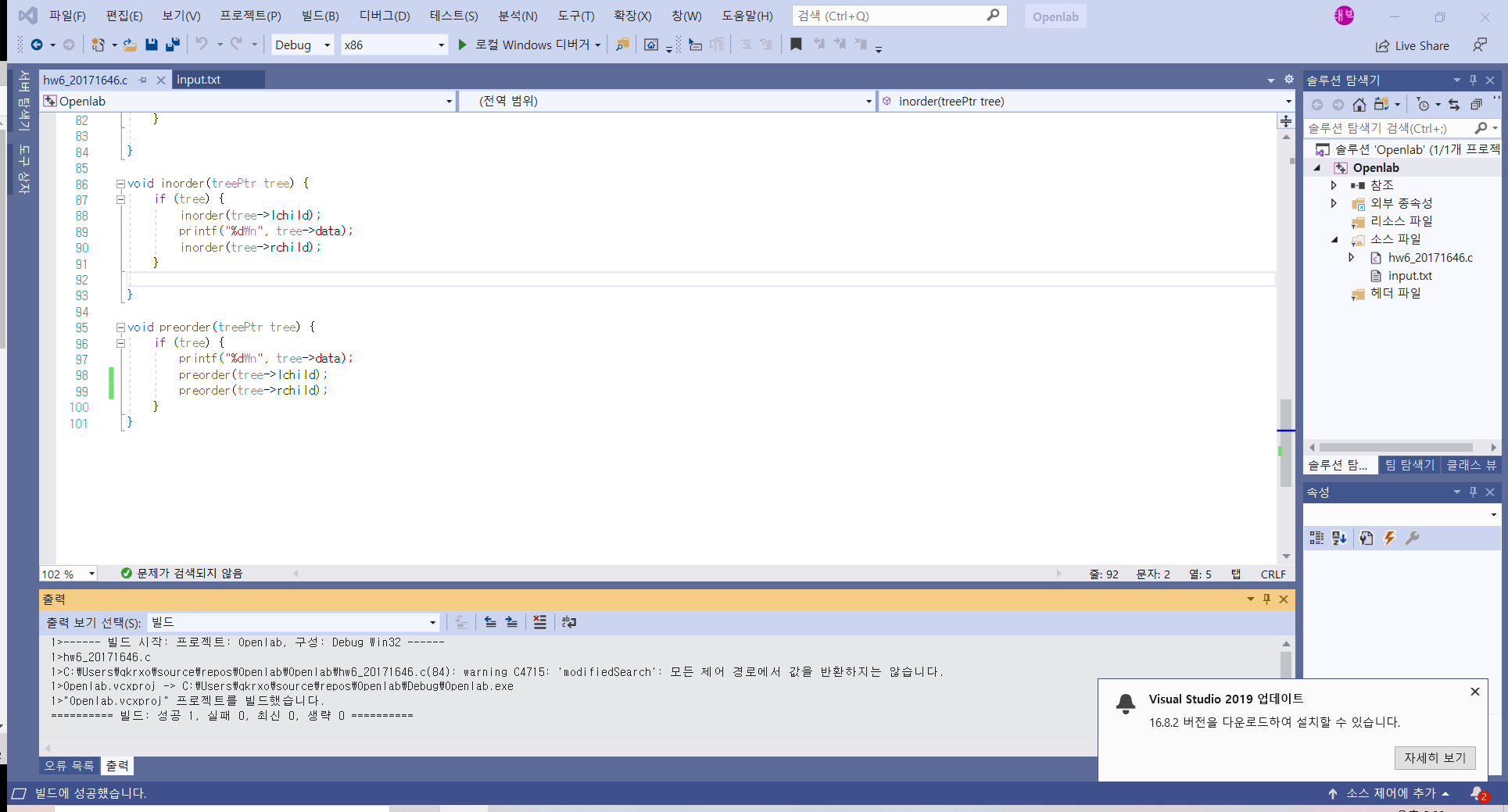


인자로 tree와 텍스트 파일에서 가져온 데이터 num을 받아 num 데이터를 가지는 트리 노드를 만든 후 트리에 연결해주는 함수이다. treePtr로 ptr과 temp를 선언하였는데 temp에는 modifiedSearch(\*tree, num)의 리턴 값을 저장하였다. tree에 이미 num데이터를 가지고 있는 노드가 연결되어 있거나 아무런 노드가 연결되어 있지 않은 경우 temp는 NULL을 나타내고 이외의 경우에 temp는 num데이터를 가지고 있는 노드가 트리에서 삽입되어야 할 위치의 부모 노드를 나타낸다. temp가 NULL이 아닌 경우 또는 !(\*tree)일 경우, 즉 tree에 아무 노드도 존재하지 않을 경우 if조건문을 실행한다. ptr을 동적 할당하여 ptr->data에 인자로 받은 num을 assign해주고 ptr의 lchild와 rchild를 NULL로 초기화 시켜준다. 이후 한 번 더 if문이 있는 것을 확인할 수 있는데, 현재 트리가 비어있는 경우와 그렇지 않은 경우를 구분한 것이다. 트리가 비어있다면 else문을 실행하고 비어있지 않다면 if문을 실행한다. if문에서 temp->data와 num을 비교하면서 ptr을 삽입하는데, 이는 temp가 현재 삽입해야 하는 위치의 부모 노드를 가리키고 있기 때문이다. temp의 데이터가 num보다 큰 경우는 temp의 lchild에 ptr을 연결하며 그렇지 않은 경우에는 temp의 rchild에 ptr을 연결해준다.

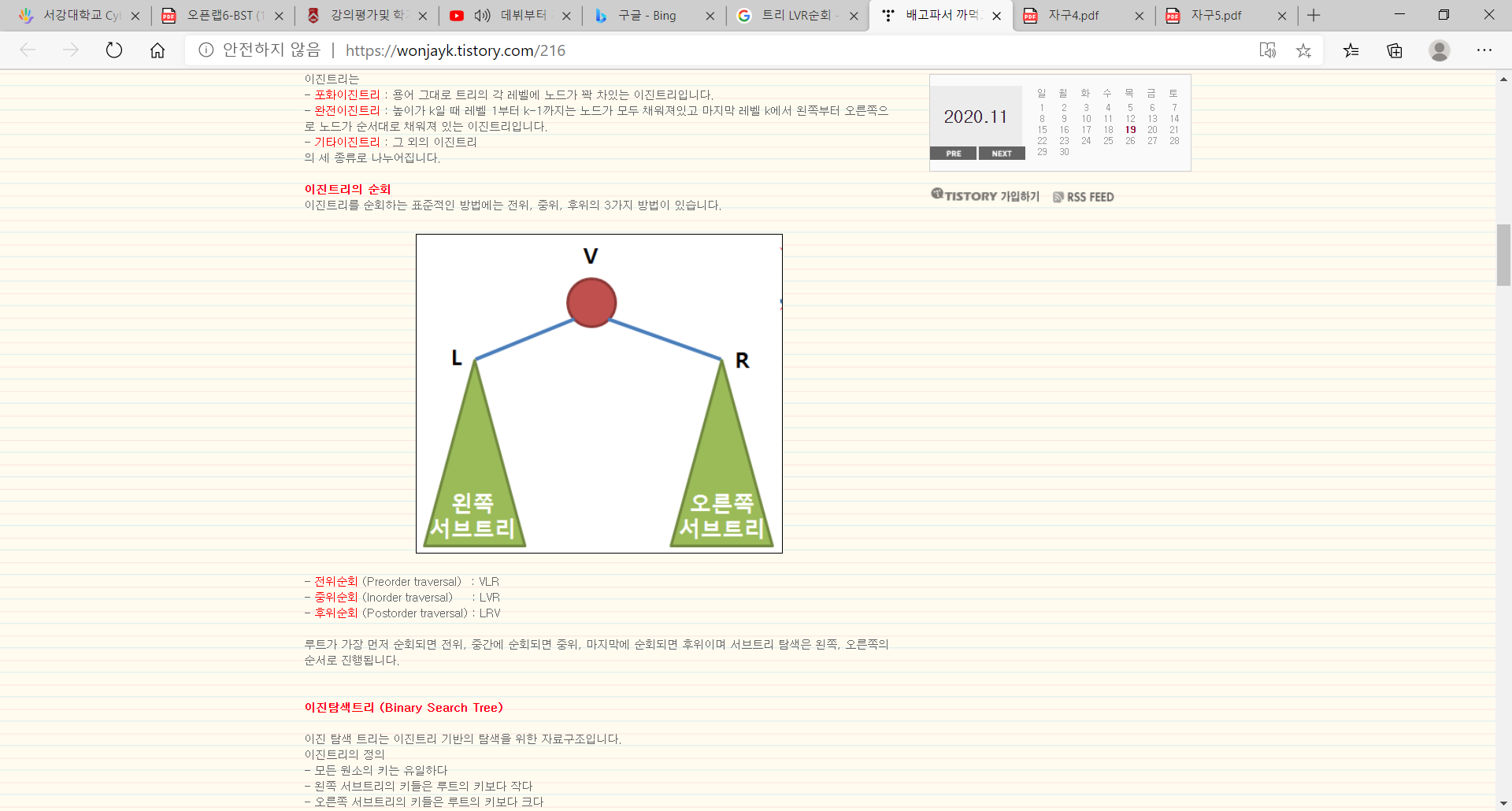
스크린샷, 노트북, 컴퓨터, 모니터이(가) 표시된 사진

자동 생성된 설명

modifiedSearch함수는 tree와 텍스트 파일에서 가져온 num을 인자로 받는다. 현재 tree에 아무런 노드도 존재하지 않는다면 NULL을 반환하며 그렇지 않다면 else문을 실행한다. treePtr temp를 생성하여 tree, 즉 root node를 처음에 가리키게 한다. 이후 temp->data와 num을 비교해가면서 temp를 트리에서 왼쪽 혹은 오른쪽으로 이동시켜주는데, 이는 재귀함수를 이용하여 구현하였다. temp를 이러한 방식으로 이동시켜가면서 num을 데이터로 가지는 트리 노드가 어느 부모 노드와 연결되어야 하는지를 알려준다. temp의 이동과정을 반복하다가 현재 temp의 데이터와 num을 비교를 하여 또 이동을 하려고 할 때 이동하고자 하는 노드가 NULL이라면 temp를 리턴해준다. 이 때 temp는 num을 데이터로 가지는 노드가 연결이 되어야 하는 부모 노드의 위치를 가리킨다. temp가 이동을 하다가 만약 temp->data == num인 경우가 생긴다면, 즉 이미 해당 데이터를 가지는 트리 노드가 이미 트리에 연결이 된 경우 else문을 실행하여 NULL을 리턴해준다.



트리를 중위(inorder), 후위(preorder)로 순회한 결과를 출력해주는 함수이다.



inorder는 왼쪽 서브트리를 순회하고 루트를 순회하며 마지막으로 오른쪽 서브트리를 순회하는 방식이며 preorder는 루트노드를 먼저 순회하고 왼쪽 서브트리를 순회하며 마지막으로 오른쪽 서브트리를 순회하는 방식이기 때문에 위와 같이 재귀적으로 구현할 수 있다.